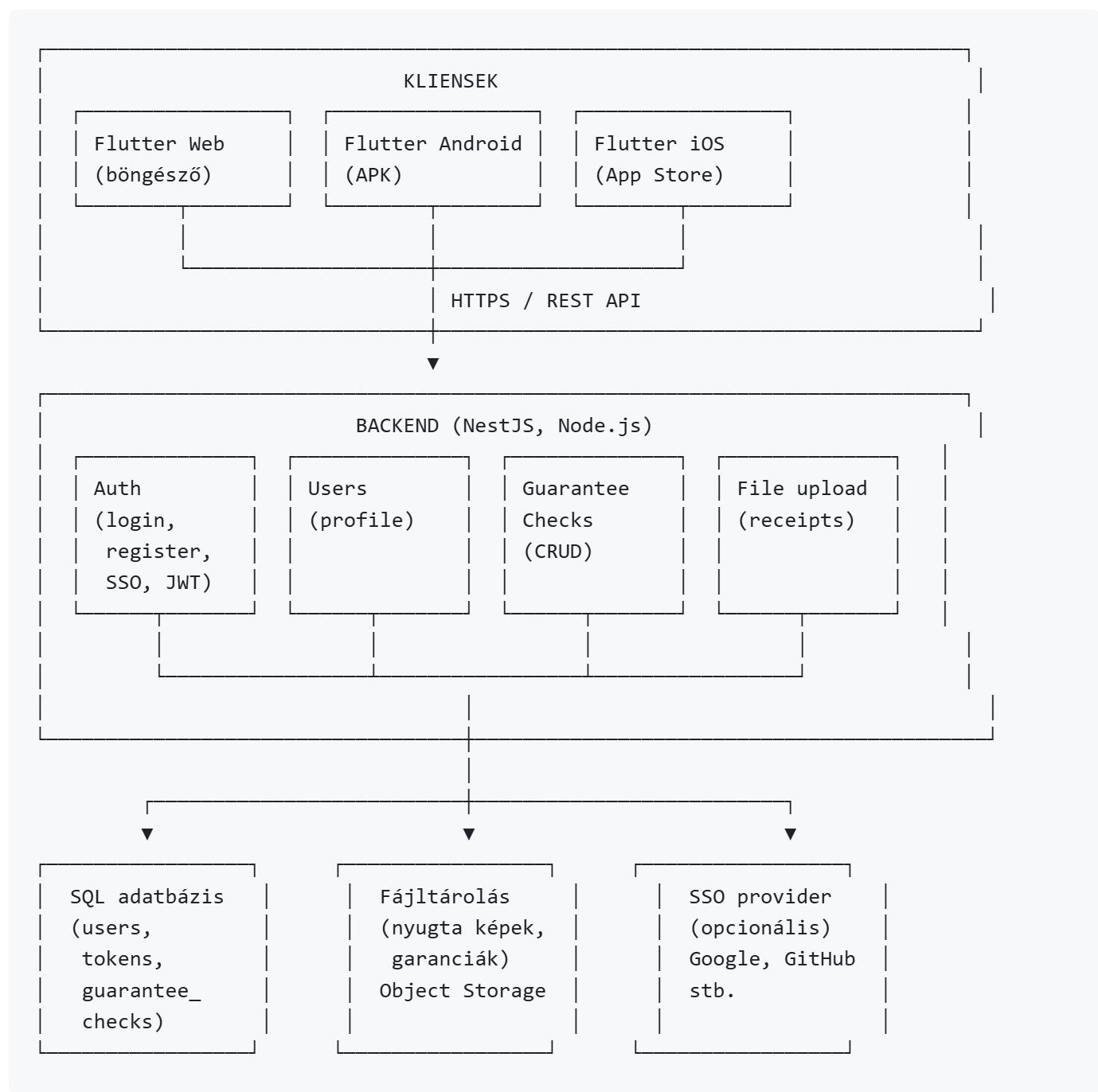


Guarantee Application – Architektúra

Ez a dokumentum a **full-stack** alkalmazás üzemeltetési és szolgáltatás-architektúráját írja le: Flutter frontend, Node.js (NestJS) backend, SQL adatbázis, fájltárolás (nyugták), valamint bejelentkezés / regisztráció / SSO.

1. Magas szintű architektúra



- **Frontend:** Flutter (web + Android + iOS) – egy codebase, több platform.
- **Backend:** NestJS (Node.js), REST API, JWT + cookie, rate limit, Helmet, CORS (SECURITY.md szerint).

- **Adatbázis:** SQL (users, refresh_tokens, guarantee_checks, stb.).
- **Fájltárolás:** Nyugták képeinek tárolása (object storage), a backend csak az útvonalat (pl. URL vagy key) tárolja az adatbázisban.
- **Auth:** Saját regisztráció/bejelentkezés + opcionális SSO (OAuth2).

1.5 Helyi fejlesztési környezet (Docker konténerek)

A teljes stack helyben **Docker Compose**-sal fut, egy parancsos indítással, a gazdagépen nem kell Node-ot vagy Fluttert telepíteni.

Szolgáltatás	Image kontextus	Port	Szerep
backend	<code>./backend</code> (NestJS, lásd <code>backend/Dockerfile</code>)	3000	API szerver (fejlesztésben in-memory DB)
frontend	<code>./frontend/guarantee_application</code> (Flutter web, nginx)	8080	Flutter web build nginx-szel kiszolgálva

Parancsok (a repo gyökeréből):

- Indítás: `docker compose up --build`
- Leállítás: `docker compose down`
- Teljes újraépítés (cache nélkül): `docker compose build --no-cache && docker compose up`

URL-ek: Frontend → <http://localhost:8080>, Backend API → <http://localhost:3000>.

Opcionális: másold a `backend/.env.example` fájlt `backend/.env`-ba, és állítsd be a `JWT_SECRET`-et (és szükség esetén a `FRONTEND_URL`-t). Részletek: [DOCKER.md](#).

2. Hitelesítés: regisztráció, bejelentkezés, SSO

2.1 Jelenlegi modell (saját auth)

- Regisztráció (email + erős jelszó), bejelentkezés, JWT access + refresh token, token blacklist, fiókzárolás.
- Ez maradhat az alap; a backend már erre épül (SECURITY.md).

2.2 SSO (opcionális) – hogyan illeszkedik

- **Cél:** „Bejelentkezés Google-lel / GitHub-bal” stb., a többi funkció (garanciák, nyugták) ugyanúgy működik.
- **Megvalósítás:** OAuth2 (pl. Passport stratégia: `passport-google-oauth20`, `passport-github2`).

„A provider (Google/GitHub) rendszerrel elválasztva a backendről a frontendtól a /auth-endpoint-en keresztül közelít.”

- A provider (Google/GitHub) reállíthat-ei vissza a backendre, a backend letervező/osszekerül egy **User** rekordot (pl. `email` + `provider` + `providerId`).
- Ugyanazt a JWT kibocsátást használod (access + refresh token), így a Flutter és a többi endpoint nem különböztet meg.
- **Alternatíva (külső IdP):** Ha nem akarsz OAuth logikát a NestJS-ben:
 - **Auth0, Clerk, Supabase Auth:** ezek kezelik a regisztrációt, bejelentkezést, SSO-t; a backend csak ellenőrzi a külső JWT-t vagy session-t, és a saját DB-ben User-t hoz létre/társít (pl. `auth0_id` vagy `supabase_uid`).
Ebben az esetben a jelenlegi „saját regisztráció/bejelentkezés” vagy marad párhuzamosan, vagy fokozatosan kivezethető.

Összefoglalva: Vagy Passport OAuth a NestJS-ben (egyszerű SSO), vagy külső IdP (Auth0/Clerk/Supabase Auth) + backend csak JWT/session validáció és user szinkron.
Mindkettő működik a te backend architektúráddal.

3. SQL adatbázis – hol érdemes futtatni

A backend jelenleg in-memory adatbázist használ; élesben **PostgreSQL** (vagy MySQL) a legcélszerűbb (tranzakciók, relációk, JWT blacklist, guarantee_checks).

Szolgáltató	Miért jó / miért nem	Ajánlás
Vercel Postgres (Neon alapú)	Jól integrálódik Vercel projekttel, serverless, PostgreSQL.	<input checked="" type="checkbox"/> Jó, ha a backendet Vercelre teszed.
Neon	Serverless PostgreSQL, ingyenes tier, könnyű.	<input checked="" type="checkbox"/> Nagyon jó önálló DB választás (Vercel nélkül is).
Supabase	PostgreSQL + Auth + Storage egy helyen; használhatod csak DB-ként is.	<input checked="" type="checkbox"/> Jó, ha később SSO-t vagy Storage-t is Supabase-ben akarsz.
Cloudflare D1	SQLite-alapú, edge, jól párosul Cloudflare Workers-sal.	⚠ SQLite; a NestJS tipikus stackje PostgreSQL. Átirás/ORM váltás kell.
Railway / Render / Fly.io	Managed PostgreSQL; klasszikus „egy VPS-szerű” backend + DB.	<input checked="" type="checkbox"/> Jó, ha Dockerrel vagy hosszú futású Node folyamattal deployolsz.
PlanetScale	MySQL, serverless.	<input checked="" type="checkbox"/> Használható, ha MySQL-t preferálsz (NestJS TypeORM/Prisma mindenkorrel).

Azure Database for PostgreSQL (Flexible Server)	Teljes körű managed PostgreSQL, régiófüggetlen árazás, VNet integráció.	<input checked="" type="checkbox"/> Jó, ha Azure-t használisz (Static Web Apps, App Service, Blob); egy felhő, egy számla.
---	---	--

Praktikus ajánlás:

- Ha Vercelre mész (backend is): Vercel Postgres vagy Neon.
- Ha Cloudflare-re mész (Workers + R2): Neon vagy Supabase PostgreSQL (D1 helyett), hogy ne kelljen SQLite-ra és edge-re átírni a backendet.
- Ha Docker / „normál” szerver: Railway, Render vagy Fly.io + beépített PostgreSQL.
- Ha Azure-ecosystem: Azure Database for PostgreSQL (Flexible Server).

4. Fájltárolás – nyugták (receipts) képei

A `GuaranteeCheck` entitásban `imagePath` van; ez a feltöltött kép **hivatkozása** (URL vagy object storage key). A tényleges fájlt object storage-ban kell tárolni.

Szolgáltató	Jellemzők	Ajánlás
Cloudflare R2	S3-kompatibilis, nincs egress díj, jól párosul Cloudflare környezettel.	<input checked="" type="checkbox"/> Nagyon jó ár/érték, ha Cloudflare-t használisz (vagy akár csak R2-t).
Vercel Blob	Egyszerű API, Vercel projekttel zökkenőmentes.	<input checked="" type="checkbox"/> Kiváló, ha frontend + backend egyaránt Vercel .
AWS S3	Ipari standard, sok dokumentáció.	<input checked="" type="checkbox"/> Mindig opció; több konfig, költség.
Supabase Storage	S3-szerű, bucketek, szerepkörök; egy helyen DB + Auth + fájl.	<input checked="" type="checkbox"/> Jó, ha már Supabase-t használisz.
Azure Blob Storage	Object storage (block blob), S3-szerű API; régió- és redundanciafüggő árazás.	<input checked="" type="checkbox"/> Jó, ha Azure-t használisz (App Service, Static Web Apps, PostgreSQL); egy felhő.

Implementációs tipp:

Backend: multipart upload végpont (pl. `POST /guarantee-checks/upload`), amely:

1. Validálja a fájlt (méret, MIME – a te `FileValidationService`-eddel).
2. Feltölte a választott object storage-ba (pl. R2 vagy Vercel Blob SDK).
3. Visszaad egy **biztonságos URL-t** vagy **storage key-t**, amit a `guarantee_checks.imagePath` -ként mentesz.

A kliens (Flutter) ezt a URL-t kapja meg a listázáshoz/megjelenítéshez; a fájl soha ne kerüljön

közvetlenül a SQL DB-be (csak a hivatkozás).

5. Deployment – hol futasd a frontendet és a backendet

5.1 Flutter frontend

Éles (production): A fő, felhasználóknak szánt frontend natív alkalmazásként kerül ki:

- **Android:** Google Play Store (APK/AAB).
- **iOS:** App Store (Xcode archive, TestFlight bétához).

Mindkettő ugyanazt a backend API-t használja (pl. <https://api.yourapp.com>).

Tesztelés / előnézet: A **web build** (`flutter build web`) **tesztelésre és belső használatra** szolgál, nem az éles fő csatorna. Hostolható pl. itt:

- **Cloudflare Pages** – ingyenes, gyors, saját domain.
- **Vercel** – statikus site ugyanabból a repóból.
- **Azure Static Web Apps** – Free tier (100 GB bandwidth, 10 app), Standard 9 USD/hó; GitHub/Azure DevOps.
- **Netlify** – hasonló.

Összefoglalva: **Flutter web** = statikusan hostolt **teszt/demo** frontend; **Flutter mobil (App Store + Play Store)** = **éles**, felhasználóknak szánt alkalmazások; mindkettő ugyanazt az API-t hívja.

5.2 NestJS backend (Node.js)

A backend **hosszú futású folyamat** (nem feltétlenül „serverless egy kérésre”); Dockerben is futtattad jelenleg.

Platform	Jellemzők	Mikor érdemes
Vercel	Serverless Functions; NestJS futtatható, de nem minden middleware/pattern illeszkedik tökéletesen; cold start.	Ha mindenhol akarsz (frontend + API), és elfogadod a serverless korlátokat.
Cloudflare Workers	Edge, serverless; a teljes NestJS nem hivatalosan támogatott (Workers más runtime).	Átírás vagy „NestJS nélküli” API kellene; nem ajánlott a jelenlegi NestJS appnak.
Railway / Render / Fly.io	„Normál” Node folyamat vagy Docker; PostgreSQL könnyen mellé.	<input checked="" type="checkbox"/> Legkezenfekvőbb a mostani NestJS + Docker + SQL stackhez.

Google Cloud Run / AWS ECS	Container alapú, skálázható.	Ha már használod a felhőt vagy nagyobb infrastruktúrát akarsz.
Azure App Service	Node.js (NestJS) futtatható; Free (F1) tier korlátozott (pl. 1 óra compute/nap), Basic/Premium folyamatosan fut.	<input checked="" type="checkbox"/> Jó, ha Azure-t preferálsz; Docker vagy natív Node deploy.

Fontos:

- **Cloudflare:** Workers helyett használd Cloudflare Pages-t a Flutter web-nek; a backendet ne Workers-re erőltessd, hanem pl. Railway / Render + Cloudflare R2 (fájltárolás) + Neon (PostgreSQL).
- **Vercel:** Flutter web + Vercel Postgres/Neon + Vercel Blob + NestJS serverless lehetséges, de a NestJS serverless konfig és cold start miatt sokan inkább Railway/Render-t választanak a backendnek.

5.3 Branch policy (Git munkafolyamat)

- **Fejlesztési alapág:** `dev`.
A feature ágak a `dev`-ből indulnak, és a `dev`-be olvadnak vissza (pl. Pull Requestekkel). A fejlesztés és a CI a `dev`-en fut.
- **Éles (production) ág:** `master` (vagy `main`).
A kód csak akkor kerül élesbe, ha a `dev`-ből merge-olva érkezik a `master`-re.
Közvetlen commit a `master`-re nem; minden előbb a `dev`-en landol.
- **Folyamat:**
`feature/xyz` → (PR) → `dev` → (PR / merge) → `master`.
Az éles deploy (App Store, Play Store, backend) kizárolag a `master`-ről történjen.
- **Opcionális:** Kötelező PR review és sikeres CI (pl. lint, tesztek) a `dev`-be merge előtt, és ugyanígy a `dev` → `master` merge előtt. A branch protection szabályok ezt kikényszeríthetik (GitHub / GitLab / Azure Repos).

6. Ajánlott kombinációk (összefoglalva)

Mindegyiknél: Flutter (web + mobile) + NestJS backend + PostgreSQL + Object storage (nyugták) + Auth (saját + opcionális SSO).

A) Egyszerű, egy hely közelében – Vercel központban

Komponens	Szolgáltatás
Flutter web	Vercel (statikus)
Backend API	Vercel Serverless (NestJS) vagy Railway (Docker/Node)
SQL	Vercel Postgres vagy Neon
Fájltárolás	Vercel Blob
Auth	Saját (meglévő) + később pl. Clerk vagy Auth0 SSO

- Ha szeretnél minden Vercelben: Vercel Postgres + Vercel Blob + NestJS serverless.
- Ha a backendet egyszerűbben akarod: Backend Railway, minden más (web, DB, blob) maradhat Vercel/Neon/Blob.

B) Cloudflare + „normál” backend

Komponens	Szolgáltatás
Flutter web	Cloudflare Pages
Backend API	Railway vagy Render (NestJS Docker/Node)
SQL	Neon vagy Supabase (PostgreSQL)
Fájltárolás	Cloudflare R2 (S3 kompatibilis)
Auth	Saját + opcionális SSO (Passport OAuth vagy Supabase Auth)

- Cloudflare: főleg Pages (frontend) + R2 (fájlok); a backend és a DB ne Cloudflare D1/Workers legyen a jelenlegi NestJS kóddal.

C) Maximálisan egyszerű, kevés szolgáltató

Komponens	Szolgáltatás
Flutter web	Cloudflare Pages vagy Vercel
Backend + DB	Railway (NestJS + PostgreSQL egy projektben)
Fájltárolás	Railway Volume vagy Cloudflare R2 / Vercel Blob
Auth	Saját (meglévő)

- Egy backend platform (Railway), egy frontend host (Pages vagy Vercel), egy SQL (Railway Postgres vagy Neon), egy object storage.

D) minden Azure-ban (egy felhő, egy számla)

Frontend	Angular
Backend	NestJS
Database	PostgreSQL
Storage	Azure Blob Storage
Auth	Azure Active Directory

Komponens	Szolgáltató
Flutter web	Azure Static Web Apps (Free vagy Standard)
Backend API	Azure App Service (Node.js / NestJS vagy Docker)
SQL	Azure Database for PostgreSQL (Flexible Server)
Fájltárolás	Azure Blob Storage (block blob)
Auth	Saját (meglévő) + opcionális Azure AD / Entra ID vagy más OAuth

- Egyetlen cloud provider: egyszerű számlázás, VNet/integráció, enterprise SLA ha kell.

7. Következő lépések (implementáció)

1. **SQL:** Válassz egy PostgreSQL szolgáltatót (Neon / Vercel Postgres / Supabase / Railway / Azure Database for PostgreSQL), kösd be TypeORM/Prisma-val, és cseréld le az in-memory DB-t.
 2. **Fájlfeltöltés:** Implementáld a backenden a multipart uploadot; válassz object storage-t (R2 / Vercel Blob / S3 / Supabase Storage / Azure Blob Storage), és mentsd a kapott URL/key-et `guarantee_checks.imagePath`-ként.
 3. **Auth/SSO:** Tartsd meg a jelenlegi JWT auth-ot; SSO-hoz adj hozzá Passport OAuth stratégiát vagy integráld egy IdP-t (Auth0, Clerk, Supabase Auth).
 4. **Deploy:**
 - Flutter web → Cloudflare Pages, Vercel vagy **Azure Static Web Apps**.
 - NestJS → Railway, Render, **Azure App Service** vagy Vercel serverless (ha azt választod).
 - Állítsd be a környezeti változókat (DB URL, storage credentials, `JWT_SECRET`, `FRONTEND_URL`).

Ha szeretnéd, a következő lépésekben konkrétan kitudjuk dolgozni pl. a **Neon + Cloudflare R2** vagy a **Vercel Postgres + Vercel Blob** integrációt a te backend kódodra (env, upload végpont, DB module).

8. Árak (díjszabás)

Az árak tájékoztató jellegűek; a szolgáltatók oldalán érvényes a naprakész díjszabás (2024–2025).

Magyarországról: A szolgáltatók általában **USD** vagy **EUR** alapú díjszabást használnak; a havi számla a bank árfolyamával HUF-ra váltva változhat. Az alábbi HUF-os becslések **1 USD ≈ 385 HUF** (2025. február) alapján készültek, csak tájékoztató jellegűek.

8.1 Frontend hosting (Flutter web – statikus)

Szolgáltató	Ingyenes / alap tier	Fizetős (rövidítve)
Vercel	Hobby: ingyenes, 200 projekt, 100 deploy/nap, 1M function invocation/hó, 100 GB fast data transfer.	Pro: 20 USD/fő/hó (~7 700 Ft) + használatalapú; 20 USD havi kredit.
Cloudflare Pages	Statikus tartalom és kérések korlátlan; Pages Functions = Workers díj.	Workers ingyenes: 100k kérés/nap; fizetős Workers díjak.
Netlify	Ingyenes tier: korlátozott build perc, bandwidth.	Pro stb. havi díj + használat.
Azure Static Web Apps	Free: 100 GB bandwidth/hó, 10 app, 250 MB tároló/app, 3 staging env, 2 custom domain, SSL.	Standard: 9 USD/hó (3–500 Ft), 100 GB bandwidth benne, 500 MB/app, 10 staging, 5 domain; túlhasználat 0,20 USD/GB (~77 Ft/GB).

8.2 Backend hosting (NestJS / Node)

Szolgáltató	Ingyenes / alap	Fizetős (rövidítve)
Vercel	Hobby: 1M function invocation/hó, 4 CPU-hour, 360 GB-hour memory.	Pro: 20 USD/fő/hó (~7 700 Ft) + használat (pl. további invocations).
Railway	Nincs örök ingyenes tier.	Hobby: 5 USD/hó minimum (~1 900 Ft) (5 USD kredit); utána használatalapú (memória, CPU, volume, egress).
Render	Ingyenes: 750 instance óra/hó, web service 15 perc inaktivitás után leáll, ephemeral fájlrendszer.	Fizetős: pl. 7 USD/hó-tól (~2 700 Ft) (web service), Postgres külön.
Fly.io	Nincs klasszikus free tier; 5 USD alatti számla nem kerül felszámításra ; új fiók egyszeri 5 USD (~1 900 Ft) kredit.	Pay-as-you-go: VM, storage, egress.
Azure App Service	Free (F1) : 10 web/API app, 1 GB tároló, 1 óra compute/nap (korlátozott); új fiók 200 USD (~77 000 Ft)/30 nap kredit + 12 hónap ingyenes szolgáltatás.	Basic/Standard/Premium: használatalapú (pl. B1 13 USD/hó (~5 000 Ft)); régiófüggő. Azure díjkalkulátor .

8.3 SQL adatbázis (PostgreSQL)

Szolgáltató	Ingyenes / alap	Fizetős (rövidítve)
Neon	0 USD: 20 projekt, 100 CU-hour/projekt/hó, 0,5 GB tároló/projekt, max 2 CU compute.	Launch: 0,106 USD/CU-hour (41 Ft), 0,35 USD/GB/hó (135 Ft). Scale: 0,222 USD/CU-hour (~85 Ft), 0,35 USD/GB/hó.
Vercel Postgres	A Vercel Postgres 2024 végétől Neonra migrált; árazás = Neon (Vercel Marketplace-en).	Ugyanaz, mint a Neon.
Supabase	0 USD: 2 aktív projekt, 500 MB DB, 1 GB fájltároló, 5 GB egress, inaktivitás után 1 hét és pause.	Pro: 25 USD/hó-tól (~9 600 Ft) (8 GB disk, 250 GB egress, 100k MAU).
Cloudflare D1	Workers Free: 5M sor olvasás/nap, 100k sor írás/nap, 5 GB tároló.	Workers Paid: 25B sor olvasás/hó benne, 50M írás/hó benne, 5 GB benne; utána 0,001 USD/M sor (0,4 Ft), 1 USD/M írás (385 Ft), 0,75 USD/GB/hó (~290 Ft).
Railway	Nincs önálló „csak DB” ingyenes.	Postgres a platformon: használatalapú (pl. volume, a Hobby 5 USD/hó (~1 900 Ft) minimummal).
Azure Database for PostgreSQL (Flexible Server)	Nincs örökl ingyenes tier; új Azure fiók 200 USD (~77 000 Ft) kredit 30 napra + egyes szolgáltatások 12 hónapig ingyen.	Pay-as-you-go: Burstable B1ms 12–15 USD/hó (4 600–5 800 Ft); tároló és IO régiófüggő. Azure PostgreSQL díjszabás .

8.4 Fájltárolás (object storage – nyugták)

Szolgáltató	Ingyenes / alap	Fizetős (rövidítve)
Cloudflare R2	10 GB tároló/hó, 1M Class A (write/list), 10M Class B (read), korlátlan egress .	Tárolás: 0,015 USD/GB/hó (6 Ft) (Standard) , Class A: 4,50 USD/M (1 730 Ft), Class B: 0,36 USD/M (~140 Ft).
Vercel Blob	Hobby: 1 GB tároló, 10k simple op, 2k advanced op, 10 GB data transfer.	Tárolás: 0,023 USD/GB/hó (9 Ft); simple op: 0,40 USD/M (154 Ft); advanced op: 5,00 USD/M (1 925 Ft); transfer: 0,05 USD/GB (~19 Ft).

Supabase Storage	Free: 1 GB fájltároló (a Free plan része).	Pro: nagyobb limit a 25 USD/hős (~9 600 Ft) plan része.
AWS S3	Nincs örök ingyenes tier (12 hónapos Free Tier lehet új fióknak).	Tárolás + kérések + egress (régiófüggő).
Azure Blob Storage	Nincs örök ingyenes tier; új fiók 200 USD (~77 000 Ft) kredit + korlátozott ingyenes szolgáltatás.	Tárolás (LRS): 0,02 USD/GB/hó (8 Ft); tranzakciók és egress külön; régió és redundancia függő. Azure Blob díjszabás .

8.5 Összehasonlító becslés (kis projektek) – USD és HUF (Magyarország)

Átváltás: 1 USD ≈ 385 Ft (tájékoztató, 2025. február).

Kombináció	Reális havi költség (nagyságrend)
Mindent ingyenesre (Flutter web + backend + DB + fájl)	0 USD (0 Ft) : Cloudflare Pages + Render free tier (backend) + Neon free + Cloudflare R2 free (vagy Supabase free 1 GB). Figyelem: Render free spin-down, Supabase/Neon free limit.
Stabil hobby (élesebb használat)	5–15 USD/hé (1 900–5 800 Ft): Railway 5 USD + Neon free (vagy marad free) + R2 free vagy Vercel Blob free; Flutter web Cloudflare Pages vagy Vercel Hobby = 0 USD.
Pro jellegű	25–50+ USD/hé (9 600–19 200+ Ft): pl. Vercel Pro (20 USD) + Neon Launch vagy Supabase Pro (25 USD) + Blob/R2 használat; vagy Railway + Neon paid + R2.
Mindent Azure-ban	0 USD (0 Ft) (új fiók): Static Web Apps Free + App Service F1 (limit: 1 óra/nap) + 200 USD (77 000 Ft) kredit 30 napra (PostgreSQL, Blob). Fizetős: Static Web Apps 9 USD (3 500 Ft) + App Service Basic + PostgreSQL Flexible + Blob → 25–50+ USD/hé (9 600–19 200+ Ft) nagyságrend (régiófüggő).

A pontos díjat mindenkor a szolgáltató aktuális díjszabása és a tényleges használat (kérések, tároló, egress) határozza meg. Magyarországról a bank árfolyama alapján a HUF költség ennél kissé eltérhet.

8.6 Árak – hivatalos URL-ek (díjszabás)

Naprakész díjak és limitek a szolgáltatók oldalán:

Kategória	Szolgáltató	Díjszabás URL
Frontend	Vercel	https://vercel.com/pricing

	Cloudflare Pages	https://developers.cloudflare.com/pages/platform/pricing/
	Netlify	https://www.netlify.com/pricing/
	Azure Static Web Apps	https://azure.microsoft.com/pricing/details/app-service/static/
Backend	Vercel (Functions)	https://vercel.com/docs/pricing
	Railway	https://railway.com/pricing
	Render	https://render.com/pricing
	Fly.io	https://fly.io/pricing/
	Azure App Service	https://azure.microsoft.com/pricing/details/app-service/windows/
SQL	Neon	https://neon.tech/pricing
	Vercel Postgres (Neon)	https://vercel.com/docs/storage/vercel-postgres/usage-and-pricing
	Supabase	https://supabase.com/pricing
	Cloudflare D1	https://developers.cloudflare.com/d1/platform/pricing/
	Azure Database for PostgreSQL	https://azure.microsoft.com/pricing/details/postgresql/flexible-server/
Fájltárolás	Cloudflare R2	https://developers.cloudflare.com/r2/pricing/
	Vercel Blob	https://vercel.com/docs/storage/vercel-blob/usage-and-pricing
	Supabase Storage	https://supabase.com/pricing (a csomag része)
	AWS S3	https://aws.amazon.com/s3/pricing/
	Azure Blob Storage	https://azure.microsoft.com/pricing/details/storage/blobs/
	Azure (kalkulátor,	https://azure.microsoft.com/pricing/calculator/ ,

Altalános	ingyenes fiók)	https://azure.microsoft.com/free/
Domain	Cloudflare Registrar	https://www.cloudflare.com/products/registrar/
	Namecheap	https://www.namecheap.com/domains/
	GoDaddy	https://www.godaddy.com/pricing
	Gandi	https://www.gandi.net/en/domain
	INWX	https://www.inwx.com/en
	domain.hu/nic.hu (.hu)	https://www.domain.hu/ , https://www.nic.hu/

9. Domain (regisztráció és hostolás)

A saját domain (pl. `yourapp.com` vagy `yourapp.hu`) a webes frontend és az API címének megjelenítéséhez kell. Két dologra van szükség: **domain regisztráció (vásárlás)** és **DNS kezelés** (a domain hová mutasson).

9.1 Domain regisztráció (vásárlás)

Itt megvásárolod a domain nevet éves (vagy többszörös éves) előfizetésre. A regisztrátor általában DNS hostolást is ad; később átviheted a DNS-t más szolgáltatóra (pl. Cloudflare), a domain tulajdonjoga marad.

Szolgáltató	Jellemzők	Árak (nagyságrend, 1 USD ≈ 385 Ft)
Cloudflare Registrar	At-cost árazás (nincs felár), ingyenes DNS + SSL + WHOIS privacy. Több száz TLD (.com, .hu, stb.).	.com: 9–10 USD/év (3 500–3 850 Ft). .hu: árfolyam függő; gyakran olcsóbb. Cloudflare domain árak .
Namecheap	Gyakori választás, gyakran első év akció. Több valutában (USD, EUR). Ingyenes WHOIS privacy.	.com: 1. év 7–11 USD (2 700–4 200 Ft), megújítás 15–18 USD (5 800–7 000 Ft). Namecheap domain árak .
GoDaddy	Nagy regisztrátor, EUR/USD, sok TLD. Az első év gyakran akciós, a megújítás magasabb.	.com: 1. év akcióval ~1–2 EUR, majd ~12–15 EUR/év. GoDaddy díjszabás .

Gandi	EU-barát, átlátható árazás, „no bullshit”. .hu és sok más TLD.	.com: 12–14 USD/év (4 600–5 400 Ft). .hu: hasonló vagy olcsóbb. Gandi domain árak .
INWX	Német regisztrátor, sok európai TLD, .hu is.	.com 10–12 USD/év , .hu ~ 6–9 USD/év (2 300–3 500 Ft). INWX .
Squarespace Domains (korábban Google Domains)	Egyszerű felület, átvétel után Squarespace kezeli.	.com nagyságrendileg ~12–15 USD/év. Squarespace Domains .
domain.hu / nic.hu	Magyar .hu domain regisztráció hivatalos / hazai szolgáltatóknál.	.hu: éves díj nagyságrendileg ~2 000–4 000 Ft (a regisztrátortól függ). domain.hu , nic.hu .

Megjegyzés: Az első év gyakran kedvezményes; a **megújítási** árat érdemes nézni hosszú távon. Cloudflare általában at-cost, ezért a megújítás is alacsony marad.

9.2 DNS hostolás (domain hová mutasson)

A domain vásárlása után a **névszerverek (DNS)** határozzák meg, hogy a domain melyik IP-re vagy szolgáltatásra mutasson (pl. Vercel, Cloudflare Pages, Azure).

Opció	Jellemzők
Regisztrátor DNS	A domain vásárlásakor adott DNS (Namecheap, GoDaddy, Cloudflare Registrar stb.). Itt állítod be az A/CNAME rekordokat a hostodra (Vercel, Pages, Azure).
Cloudflare DNS (ingyenes)	A domain „proxy”-zod Cloudflare-en: a domain a Cloudflare-ra mutat, onnan továbbítja a Vercel/Pages/saját szerver felé. Ingyenes DNS, SSL, DDoS alapvédelem. A domain regisztráció maradhat másolat (Namecheap, Gandi), csak a névszervereket állítod Cloudflare-re.
Vercel / Cloudflare Pages / Azure	A host oldalán add hozzá a custom domainet; a felület elmondja, milyen A vagy CNAME rekordot állíts be (pl. CNAME app → cname.vercel-dns.com). SSL általában ingyenes (Let's Encrypt vagy a platform).

Gyakorlatban (Magyarországról): Vásárolj domainet Cloudflare Registrar-nál vagy Namecheap/Gandi/INWX/domain.hu-nál → állítsd be a DNS-t a választott hostra (Vercel, Cloudflare Pages, Azure Static Web Apps). Ha Cloudflare-t használsz (Pages vagy csak DNS), add hozzá a domainet a Cloudflare dashboardon, és kapj ingyenes DNS + SSL-t.

9.3 Összefoglalás – domain költség (éves, tájékoztató)

Típus	USD/év (nagyságrend)	HUF/év (~385 Ft/USD)
-------	----------------------	----------------------

.com	~9–18 (regisztrátortól és 1. év vs. megújítás)	~3 500–7 000 Ft
.hu	~6–10 (vagy hazai Ft árazás)	~2 300–4 000 Ft
DNS hostolás	0 (regisztrátor vagy Cloudflare DNS ingyenes)	0 Ft

A pontos árak a regisztrátor oldalán és a választott TLD alapján érvényesek; Magyarországról a bank árfolyama befolyásolja a tényleges HUF költséget.

9.4 Domain – hivatalos / árazó linkek

- [Cloudflare Registrar](#) – at-cost regisztráció, ingyenes DNS/SSL
- [Namecheap](#) – domain árak
- [GoDaddy](#) – díjszabás
- [Gandi](#) – domain
- [INWX](#)
- [domain.hu](#) – .hu domain (magyar)
- [nic.hu](#) – hu regisztráció